

AGREEMENT TO DEVELOP MOBILE APP

THIS AGREEMENT ("Agreement") is between Software Developers Inc. ("SDI"), a Delaware Corporation having its office at 18809 Cox Avenue, Ste. 100, Saratoga, CA - 95070, USA and Jody Rogers, 1127 North California Street, Burbank CA 91505 ("Client"). This Agreement shall be deemed to be dated and effective as of the date the last of the parties signs the Agreement (the "Effective Date").

1. In consideration of the work to be performed by SDI as herein described, Client shall pay to SDI the amount of \$25,700, in two installments of \$12,850, subject to the terms and conditions set forth herein. The first installment shall be paid within two (2) business days following the Effective Date of this Agreement. Unless otherwise excused (e.g., subject to the provisions of Paragraph 5, below), Client shall pay the second installment of \$12,850 with 10 business days of receipt of the working application.

2. SDI undertakes to develop a mobile application for Client pursuant to the specifications contained in the document entitled Scope of Work: HavRez App, v.1.4, which constitutes the entire scope of work (the "SOW"). (A copy of the SOW is attached hereto as Exhibit 1 and is incorporated herein as though set forth in full at this place.) The SOW was reviewed and approved by the parties on the Effective Date of this Agreement. Any specifications and/or details not mentioned in this Agreement or the SOW, or the Non-Disclosure Agreement, are not valid and are not to be deemed included in this Agreement. (A copy of the Non-Disclosure Agreement is attached hereto as Exhibit 2 and incorporated herein as though set forth in full at this place.) If the Client desires additional features, functionality, pages or tasks and/or provides any variation to the agreed specifications, then these will be considered as change request(s) or additional enhancements. SDI shall advise promptly if it will perform these changes, and if so, if SDI will charge an additional amount for such change(s) and how such additional charge, if any, is determined. Unless the requested change is agreed to by the parties (both as to the change itself and the cost), SDI is not obligated to complete change requests or changes outside of the SOW.

3. If the Client requires SDI to provide design services, Client shall advise SDI with written instructions on Client's design expectations if any. Within 10 business days of SDI's receipt of Client's written instructions, SDI will advise if SDI will provide the requested services and if there shall be a cost associated with SDI's performance of said services. If the parties agree that SDI will provide design services, such design services shall include a maximum of up to 3 design iterations as needed to meet Client's expectations. Except in the event of SDI's negligent or willful misconduct, SDI shall not be liable to the Client if such designs do not meet with the Client's approval within these 3 iterations. If the parties wish to proceed with further iterations, and the parties do not agree to a different process, the process shall be as set forth above.

4. SDI follows design and programming standards as per two internal documents, collectively referred to as "SDI Design and Programming Standards" ("SDI Design Standards"). A copy of each of the two documents which, together constitute SDI Design Standards, are attached as exhibits hereto, to wit, a document entitled "iOS Design Standards for SDI Apps" and a document entitled "iOS Coding Standards for SDI Apps" are attached hereto as Exhibits 3

and 4, respectively. Both documents are current as of the Effective Date. If Client requires any specific standards of design or programming which are different from those contained in the SDI Design Standards, then Client must provide those other standards and/or requirements in a detailed document before the start of the project and SDI shall then decide to accept or reject the project with 5 business days of SDI's receipt of Client's request. Client and SDI shall agree with regard to the provisions of this paragraph before Client is obligated to SDI for the contract price or payment of the first installment or any other obligations of SDI. (A copy of the SDI Design Standards is attached hereto as Exhibit 2, and is incorporated herein as though set forth in full at this place.)

5. The app produced and delivered by SDI to Client may contain "bugs" or problems in functionality or delivered features unknown to SDI. If SDI is notified by Client in writing about "bugs" in the app for a period up to 180 days from the date of delivery, then SDI will make all efforts to resolve the "bugs" and a solution will be sent to Client. The bug resolution services will be provided with reasonable skill and care in accordance with usual industry practice and in a timely, workmanlike and effective manner, not to exceed 180 days from receipt of notice from Client. If SDI fails to cure the problem with said 180 days, Client shall not be obligated to pay the second installment due under this contract or any other outstanding sums.

6. SDI will test the app on any 2 popular devices (iOS and/or Android as applicable). If Client wants the app to be tested on more than 2 devices or on multiple levels then SDI may apply an additional fee at its sole discretion, provided that SDI first advises Client of the cost and Client agrees to pay same.

7. All apps are developed as Native apps or Framework apps. Native apps will be created via the SDKs ("Software Development Kit") provided by the principal company—Apple Inc. for iPhone/iPad apps, Google Inc. for Android apps, or as per the 3rd party SDK that is employed for the service. SDI's responsibility is to provide a working app under the latest version of the SDK at the time of delivery. If Client wishes to make the app compatible with future versions of SDKs, SDI may charge an additional fee as and when Client requests compatibility with the latest version of the SDK. For framework apps, SDI may use ready-made code, ready-made CMS ("Content Management System") and/or ready-made Frameworks in addition to custom created code. These ready-made codes may be used to expedite project development and will be used at SDI's discretion as deemed feasible for the project, provided however, that SDI first provides Client with prior written notice that such codes may be used, and SDI receives Client's written approval. Failure of Client to provide written approval shall be deemed a refusal to use such ready-made codes.

8. This Agreement shall not be cancelled by either party, except with the written consent of both parties or as otherwise provided in this Agreement (e.g., paragraph 4, above).

9. SDI will provide regular updates which will include statements of WIP ("Work in Progress"). Each statement will successively outline the completed work in percentage.

10. In the event that Client fails to supply information or instructions within 30 business days of an email or written request from SDI and SDI is thereby unable to perform its obligations under the Agreement, SDI shall be entitled forthwith to terminate this Agreement. In

the event of such termination, the Client shall owe SDI an amount equal to the percentage of the work completed as set forth on the most recent WIP. Thus, by way of example, if SDI terminates at 60% of work completion and the Client has already paid 50% of the project cost, then the client will owe SDI the balance 10% of the project cost. In consideration of the physical payment from Client to SDI of the balance due, SDI will deliver to Client all completed work and work in process as of the date of the exchange.

11. SDI will provide services following standard business and professional practices and ethics. If Client engages in ongoing rude, improper or abusive language or behavior in ongoing communication with SDI or its employees unrelated to the development of the app, then SDI reserves the right to refuse to provide its services after providing written notice to Client of the offending conduct and Client is afforded a reasonable opportunity to cure or otherwise refrain in the future from such conduct. In such an event, the Client shall have no further financial obligations to SDI and SDI shall promptly return to Client all completed work and all work in process.

12. Other than for good cause, if Client requests cancellation of a contract before the work is completed, Client agrees to pay SDI, and SDI shall deliver all completed work and work in progress based on the formula and method of exchange as provided for in paragraph 10, above. If for good cause, the Client shall have no further financial obligations to SDI and SDI shall promptly return to Client all completed work and all work in process.

13. If not for good cause, then Client shall be obligated for any damages to SDI not to exceed \$12,850 (i.e., the balance of the total consideration paid by the Client). The effective date of cancellation is to be 30 days from the date of the written request for cancellation.

14. Intentionally Deleted.

15. Except in the case of SDI's negligence or willful misconduct, neither Client on the one hand, nor SDI (including its subsidiaries, affiliates, officers, and employees) on the other hand, shall be liable to the other for any direct, indirect, incidental, special, consequential, exemplary or punitive damages, including but not limited to, damages for loss of profits, goodwill, use, data, or other intangible losses (even if such party has been advised of the possibility of such damages). Such limitation of liability shall apply whether the damages arise from the use of or inability to use SDI's services, reliance on SDI's services, or from the interruption, suspension, or termination of SDI's services (including such damages incurred by third parties). This limitation shall also apply to the costs of procurement of substitute goods or services resulting from products or services purchased or obtained or messages received or transactions entered through SDI's services or for unauthorized access to or alteration of Client's data or transmissions and any statements or conduct of a third party or any other matters relating to SDI's services.

16. Client agrees to defend, indemnify and hold harmless SDI, its directors, officers, employees and agents from and against all claims and expenses, including attorney fees that may arise or result from any content Client submits, posts, transmits or makes available through SDI's services, from any product sold by Client, its agents or employees, from any service provided or performed or agreed to be performed by SDI or from Client's breach or violation of this

Agreement, including any obligation, representation, or warranty made herein by Client, or Client's violation of any rights of another. Client further agrees to defend, indemnify and hold harmless SDI, its directors, officers, employees and agents from and against all claims and expenses, including attorney's fees, arising from or related to contracts, representations, agreements, promises, etc., made between Client and third parties, or arising from or related to Client's negligence toward third parties. NOTWITHSTANDING THE FOREGOING, UNDER NO CIRCUMSTANCES WILL CLIENT BE LIABLE TO SDI IN EXCESS OF THE SUM OF TWENTY FIVE THOUSAND, SEVEN HUNDRED FIFTY DOLLARS (\$25,700) EXCEPT IN THE CASE WHERE THE SOW HAS BEEN AMENDED IN A SEPARATE WRITING REVISING THE PAYMENT OBLIGATIONS OF CLIENT AND SIGNED BY CLIENT. IN SUCH CASE CLIENT SHALL ONLY BE LIABLE TO THE EXTENT OF A CHANGE IN CLIENT'S MONETARY OBLIGATIONS SET FORTH IN SUCH AMENDMENT.

17. Client agrees that it shall not during the continuance of this Agreement and for a period of 3 years following the expiration or termination of this Agreement (however arising) employ, solicit or contract the services of any person or independent contractor who is or was employed or engaged by SDI. SDI will be assigning a team of the following personnel: Project manager, Senior Designer, Senior iOS Programmer, Senior Android programmer, Web services leader and Tester. SDI will provide individual names within 10 days of receiving the signed contract.

18. SDI reserves the right to subcontract services or assign the ongoing servicing and/or hosting of your account or this entire Agreement to another party at its sole discretion, but only following written notice to Client of the specifics and Client's written approval of same. Any such assignment or subcontracting will not relieve SDI of its obligations under this Agreement. This Agreement shall not be affected by any change in the name of SDI and shall continue in effect thereafter in accordance with its terms.

19. All notices or requests required or permitted by this Agreement shall be in writing and in English and may be delivered personally, or may be sent by email or certified mail, return receipt requested, to the address set forth at the end of this Agreement. If either party chooses to send a notice or a request by email, the receiving party shall confirm receipt of the email. Confirmation of receipt is not to be deemed agreement or rejection of the content of the email but is intended solely to confirm receipt. If no written confirmation of receipt (e.g., by email) is received within 2 business days following the sending of an e-mail, a copy of the notice or request will only be deemed effective if sent by messenger or overnight courier.

20. This Agreement shall be governed exclusively by the laws of the State of California, USA, without regard to any conflicts of law provisions thereof, as a contract entered into and performed entirely within the State of California. The parties hereby expressly disclaim the application of the United Nations Convention on the International Sale of Goods. The parties agree that in lieu of litigation, arbitration shall be used as the means of resolving disputes. Arbitration shall be through a neutral third-party arbitrator to be approved by both Client and SDI. The decision of the Arbitrator will be binding on Client and SDI. If the parties cannot agree on an arbitrator, then either party may commence litigation in order to have the matter resolved by arbitration. Said litigation shall only be commenced in the State of California. The prevailing

party in any litigation or arbitration between the parties pertaining to this Agreement may be entitled to an award of attorney fees and costs incurred.

21. SDI's full and complete liability, if proven, for any reason whatsoever except for negligent or willful misconduct or the Non-Disclosure Agreement between the parties, shall be limited to the full refund of all monies paid to SDI by Client. (A copy of the Non-Disclosure Agreement is attached hereto as Exhibit 3, and is incorporated herein as though set forth in full at this place.)

22. Contact Information: Client may contact SDI by phone at 408.647.2206 Monday through Friday (U.S. Working days) from 9:30 a.m. to 3:30 p.m. Pacific Standard Time. Client may also email SDI for general questions at team@sdi.la. Other requests can be sent by mail to:

Attn: Software Developers Inc., 18809 Cox Avenue, Ste. 100, Saratoga, CA, 95070.

23. Contact Information: SDI may contact Client by phone at 818. 818.967.7071, Monday through Friday (U.S. Working days) from 9:30 a.m. to 3:30 p.m. Pacific Standard Time. SDI may also email Client for general questions at jodyrogers11@att.net. Other requests can be sent by mail to: Ms. Jody Rogers, 1127 North California Street, Burbank CA 91505.

24. As used herein, "Work Product" means all HTML files, Java files, graphics files, animation files, data files, technology, scripts, and programs, both in object code and source code form, all documentation, and all other items and information, whether tangible or intangible and in whatever form or media, created, written, conceived, made, or discovered by SDI or any SDI personnel or independent contractors in connection with the performance of this Agreement.

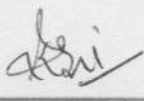
25. As used herein, "Intellectual Property Rights" means, on a worldwide basis, any and all now known or hereafter known tangible and intangible (a) rights associated with works of authorship including, without limitation, copyrights, moral rights, and mask-works, (b) trademark and trade name rights and similar rights, (c) trade secret rights, (d) patents, designs, algorithms, and other industrial property rights, (e) all other intellectual and industrial property rights of every kind and nature and however designated, whether arising by operation of law, contract, license, or otherwise, and (f) all registrations, initial applications, renewals, extensions, continuations, divisions, or reissues thereof now or hereafter in force (including any rights in any of the foregoing).

26. The Work Product is and shall remain the sole and exclusive property of Client, and Client shall retain all Intellectual Property Rights therein. If SDI is deemed to retain any Intellectual Property Rights in any Work Product under applicable law, SDI hereby irrevocably assigns to Client all such Intellectual Property Rights. If SDI has any such Intellectual Property Rights that cannot be assigned to Client under applicable law, SDI waives the enforcement thereof. If SDI has any such Intellectual Property Rights that cannot be assigned or waived under applicable law, SDI hereby grants to Client an exclusive, worldwide, sub-licensable (through multiple tiers), assignable, royalty-free, perpetual, irrevocable, fully paid-up license to use, reproduce, distribute (through multiple tiers), create derivative works of, publicly perform, publicly display, digitally perform, make, have made, sell, offer for sale, and import such Work Product. SDI acknowledges that there are, and may be, future rights that Client may otherwise

become entitled to with respect to the Work Product that do not yet exist, as well as new uses, media, and means and forms of exploitation throughout the world exploiting current or future technology yet to be developed, and SDI specifically intends the foregoing assignment of rights to Client to include all such now known or unknown uses, media, and means and forms of exploitation throughout the world.

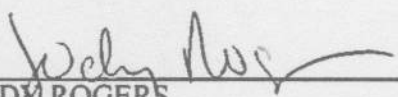
Dated: April 3, 2015

SOFTWARE DEVELOPERS INC.

BY: 

SAKSHI SHARMA

Dated: April 3, 2015

BY: 

JODY ROGERS

AGREEMENT

**Between
Software Developers Inc. ("SDI")
and
Jody Rogers**

EXHIBIT 1

sdi

SCOPE OF WORK: HARVEY APP

v.1.4

Confidentiality of Important Information

The information in this Document is confidential and is intended solely for the attention and use of Ms. Jody Rogers. It might contain privileged information. If it has come to you in error and you are not the intended recipient you must not proceed further, disclose, copy, use or disseminate any information contained therein, please delete it and contact us (sakshi@sdi.la) without delay so that we may take whatever action we consider appropriate. Although this document is believed to be free from any virus it remains the responsibility of the recipient to ensure that this document is virus free and we accept no responsibility in this regard.

Table of Notifications

This Table represents the contacts in both companies as assigned currently.

Project	Name	Company
Project Requirements	Jody Rogers	HavRez
Client Interface (Business)	Sakshi Sharma	Software Developers Inc.
Client Interface (Technology)	Raj Srivastav	Software Developers Inc.

Aim

- To build a restaurant finder app which will allow users to find a restaurant based on their preferences (Profile) instantly in real time. This app will be modeled on Cognitive and Standard behavior.
- The users will be provided an algorithmic result based on answers to some question and based on their tastes and preferences the app will recommend restaurants.
- The users can select a restaurant from the recommended results. Optional features like reservations may be available. They can also skip the recommendations and start search again.
- This document contains all of the suggested main modules/functions requirements.

Development Principles

- Mobile app for iOS platform (iPhone)
- Combination of latest UI and UX principles to provide a clean, Intuitive Interface for the HavRez app
- Smart Navigation Tabs for easy and fast access to all functions
- Language of Development – English
- Language of Data Entry - English

Coding Standards

- Apple iOS SDK's will be used to create the code and user interface design.

Requirements from client

- Client will provide upto 50 questions which can be in a series or in sets.
Note : The client can provide these in an excel spreadsheet or can directly add these to the database through the backend interface.
- We will create an algorithm/logic which will pop-up questions based on users responses (Cognitive behaviour)

FEATURES/ FUNCTIONS:

MODULE 1 : App Features/Functions

1. Sign In with your Email and Password or connect via your Facebook account.
2. Sign Up if the user does not have a Facebook account
 - 2.1. Name
 - 2.2. Email
 - 2.3. Password
 - 2.4. Confirm password

Note : First time users will have an option to skip the sign up step to check out the app.
3. **Complete profile:** After the user signs up, "Complete your profile" screen populates. It will help the app to understand the user's basic tastes and preferences. These will be provided by the client and we will include them into the app/back-end.
4. Find a Restaurant

4.1. Click on the "Find a restaurant" button which will start a round of questions.

4.2. User will select a response and move to the next question.

Note: The users cannot skip the questions. They have to provide a response in order to move forward. The questions will be based on an algorithm which will be sequenced based on the aspirational profile, the next question will appear only after and if it is needed based on the previous answer.

4.3. The app will provide an option to the users to select a radius within which they would like to dine in. For example 5miles, 10miles or 30miles.

Note : The app will recommend restaurants based on the users geolocation and selected criteria.

4.4. A list of restaurants will pop up as soon as the questions round is over. Each restaurant will also display an image next to its name. The list may consist of 3 different restaurants. If users are not satisfied with the best recommended restaurants then they can swipe to the next best recommendations.

4.5. The recommended restaurants will provide following pieces of information:

4.5.1. Restaurant name

4.5.2. Attributes (which will be pulled from the database, these attributes are going to be 2 best dishes of that particular restaurant)

The restaurants database can consist of unlimited restaurants (List of restaurants to be provided by the client in a spreadsheet) and upto 50 attributes can be associated with each restaurant.

4.6. Once the user selects a restaurant the following information shows up:

4.6.1. Image icon

4.6.2. Address (map integration to get directions)

4.6.3. Contact number

4.6.4. Cuisine type

4.6.5. Reservation option to make a reservation

We will integrate up to 3 APIs from different providers if needed to provide easy restaurant reservations.

- 4.7. If the app couldn't generate results based on users response then a message will popup saying "Sorry, we couldn't find any restaurants based on your preferences."
- 4.8. The next screen after the above message will display a default list of restaurants nearby users current location (users can change the location if they would like to search for restaurants in a specific location)

Note: The app will show results based on the backend database.

- The app will pull the GPS location data of the user, convert the data into geo-coordinates & then query the location field in the restaurants database table.
- It will then filter the above results by comparing the user data stored in the user profile database table & user preference attributes database table (quiz answers fields, food preference field etc) with the restaurants attributes (rankings field (based on the quiz), cuisine field) database table to provide the results as per users profile.

We will feed the database in a way that users will not be able to search restaurants for locations that do not exist in our database. We will acknowledge their request with a message popup "Sorry, we do not serve this location at the moment".

- 4.9. Trapdoor - If a user visits or is in the vicinity of the restaurant and decides to not go/eat at the restaurant, then he can click on the "Trapdoor" icon and then he will be presented with another set of recommendations. If the visitor decides to visit another non-recommended restaurant then he/she will have the ability to add that restaurant to the app
- 4.10. Favourites - User will be able to mark restaurants as "Favourites" and he will be able to view the Favourites list.

5. Settings

- 5.1. Edit profile
- 5.2. Edit preferences
- 5.3. Change password
- 5.4. Delete account

5.5. Signout

6. Help/ Info

MODULE 2 : Web Services

App Features/Functions for the Admin

1. Login/ Logout
2. Dashboard
3. Add/edit/delete questions {The algorithms (understanding and creating a logic in terms of what will be the next question and matching the results) will be created and the code will be written accordingly in order to find the restaurant}
4. Manage and control users access to database.
5. Add/ Modify/ delete accounts/ users profile.
6. Users can submit a restaurant and also send comments.
7. Customer loyalty module.

Other salient points

- We will launch the app on a local (Specific cities) basis initially based on the clients preferences. As more data becomes available, more cities can be added.
- The app will have an integrated Flurry based analytics. This will provide insights into user behavior and app usage data.
- We will provide a 3 -4 page static website for company contact information and app features. This is a requirement to be fulfilled before launch as per Apple's SDK guidelines.
- In-App payments can be integrated at a later date as an optional feature.
- We will focus on designing the app to increase loyalty.
- The design will reflect familiarity with other popular restaurant apps for easy adoption.
- The app will have geo-location identification.

Investment details - Time & Cost

Tasks	Resources	Duration	Cost
Wireframes and designs	1 Designer	1 month	\$3200
iOS App development	1 iOS developer	3 months	\$12000
Backend + Web services	1 PHP programmer	3 months	\$10500
Testing	1 QA Engineers	1 month	\$0
Beta Launch		1 week	\$0
		TOTAL	\$25700

FYI: Testing and beta launch are the final stages of project completion where we will test the app and make sure that everything is working smoothly without any issues. There is no charge for these services. We have listed this to keep you informed about the time it will require.

Payment and revenue sharing terms:

- 50% upfront payment at contract signing.
- 50% on Beta delivery.

Project Management - Collaboration/Review/Feedback

- We use Active-Collab for Project Management/communications. You will be provided with login credentials and you can provide your feedback, information, and/or ask questions if any. You can communicate with the team of designers and developers who will be working on your project.
- The process will start by going through the scope of work with team. We will provide you with a project plan which will include a list of tasks with their scheduled completion deadlines.
- The project manager will have a meeting with you when you are ready and we will begin work on the wireframes and design work and thereafter regularly update you and receive feedback and suggestions until you approve the designs.

- Once the designs are approved, we will move to the development phase. The team will work on the frontend and backend in parallel. We will have regular meetings and schedule the meetings as per your convenience.
- You will have daily access via phone, email, skype, and gotomeeting to directly communicate with the Project manager and/or other team members.
- When the team starts testing the app, we will provide you the build which you can test on your respective device. We will ask you to share your iPhone device UDID. You can provide your additional UDIDs as well to test the app among your friends and relatives before it goes live.
- All deliveries are covered by a 6 months debugging warranty after delivery.

AGREEMENT

**Between
Software Developers Inc. ("SDI")
and
Jody Rogers**

EXHIBIT 2

CONFIDENTIAL DISCLOSURE AGREEMENT

This AGREEMENT (the "Agreement") is made and entered into on Nov 13, 2014 by and between [I have Reservations] having an office at [1127 N. California Street, Burbank, California 91505] ("Company") and Software Developers Inc. having an office at 18809 Cox Avenue, Suite 100, Saratoga, CA 95070 ("Vendor"), hereinafter collectively referred to as the "Parties."

Company is exploring the possibility that Vendor may be able to provide information technology services to Company. In the course of such discussions, the Parties expect to disclose confidential information to each other. The documentation and content of such discussions and such disclosures are proprietary and confidential to each party. The Parties agree that the disclosure of information defined below as Confidential Information shall be governed by this Agreement.

NOW, THEREFORE, for good and valuable consideration, the Parties mutually agree as follows:

1. "Confidential Information" shall mean nonpublic information revealed by or through a party (whether in writing, orally or by another means) (the "Disclosing Party") to the other (the "Receiving Party") including (a) information expressly or implicitly marked or disclosed as confidential, (b) information expressly or implicitly marked or disclosed as proprietary trade secrets, (c) Company's Request for Proposal (the "RFP"), as issued and supplemented; Vendor's proposal in response to the RFP, and as supplemented, and information disclosed by both Parties in the process of evaluating Vendor's proposal, including but not limited to due diligence plans provided by Vendor, due diligence information supplied by Company; terms sheets and information disclosed in the completion of the term sheet evaluation process; and draft agreements and schedules thereto; (d) all forms and types of financial, business, scientific, technical, economic, or engineering information including patterns, plans, compilations, program devices, formulas, designs, prototypes, methods, techniques, processes, procedures, programs, or codes, whether tangible or intangible, and whether or how stored, compiled, or memorialized physically, electronically, graphically, photographically, or in writing, which is identified with the legend "confidential", "restricted", "proprietary", or with a similar designation and (e) all copies thereof.

2. The disclosure of the Confidential Information hereunder is for the sole purposes of the mutual exploration of Vendor potentially providing ideation, marketing, information technology services and all intellectual property related to endeavor. As to any Confidential Information disclosed by the Disclosing Party to the Receiving Party, pursuant to paragraph 1 hereof, the Receiving Party shall take reasonable precautions in accordance with procedures it follows with respect to its own important confidential information to prevent disclosure, directly or indirectly, of all or any portion of the Confidential Information.

3. The Receiving Party agrees not to otherwise use the Confidential Information obtained hereunder in the absence of a written agreement with Disclosing Party. Neither party shall disclose, without the prior written consent of the other party, the fact that discussions or negotiations are taking place with respect to the possible business relationship and the RFP. For avoidance of doubt, the Receiving Party shall not disclose or disseminate Confidential Information disclosed by or on behalf of, or related to, the Disclosing Party to any potential

partner or subcontractor without first obtaining (a) the written consent of the Disclosing Party, and (b) an executed nondisclosure agreement with such partner or subcontractor, enforceable by the Disclosing Party, under which such partner or subcontractor agrees to abide by the terms to which the Receiving Party is bound under this Agreement. Upon the Disclosing Party's request, the Receiving Party agrees to return or destroy, as the Disclosing Party may direct, all material in any medium that contains or discloses Confidential Information disclosed by or on behalf of such Disclosing Party, and retain no copies; provided however, the Receiving Party may retain one copy of such Confidential Information solely for the purpose of addressing legal claims. The Receiving Party also agrees to provide the Disclosing Party on request with a written certification that such return and destruction has been completed. Nothing in this Agreement precludes either party from using General Knowledge in conducting its business activities. "General Knowledge" means general know-how, ideas, concepts or techniques related to information technology and included in the discussions and disclosures made herein that are retained in the unaided memories of the employees of either party who have had access to information consistent with terms of this Agreement. An employee's memory is unaided if the employee has not intentionally memorized the information for the purpose of evading obligations contained in this Agreement. All General Knowledge is subject to all valid patents, copyrights and trade secrets. Nothing in this provision shall give either party the right to disclose, publish or disseminate the source of General Knowledge or the financial, statistical or personal data or business plans of the other party.

4. The obligations under paragraphs 2 and 3 hereof remain in full force and effect until and unless: (a) the Receiving Party can show that such Confidential Information was in the Receiving Party's possession prior to the date of the disclosure by Disclosing Party; or (b) such Confidential Information was obtained by the Receiving Party after the date of this Agreement from a party other than Disclosing Party, said party being under no obligation of confidentiality to the Disclosing Party with respect to such information; or (c) such Confidential Information was disclosed by the Disclosing Party to a third party without obligation of confidentiality or otherwise becomes generally available to the trade, or to the public, based on existing records or which becomes generally available to the trade or to the public through sources other than Receiving Party; or (d) such Confidential Information is developed at any time by the Receiving Party independent of Confidential Information disclosed by Disclosing Party to the Receiving Party.

5. In the event that the Receiving Party is requested or required by a governmental entity (by oral questions, interrogatories, requests for information or documents, subpoena, civil investigative demand or similar process) to disclose any Confidential Information furnished by the Disclosing Party, it is agreed that the Receiving Party will, to the extent permitted by law, cooperate with the Disclosing Party and provide the Disclosing Party with prompt notice of such request(s) or requirement(s) so that the Disclosing Party may seek an appropriate protective order or waive compliance by the Receiving Party with the provisions of this Agreement. If, in the absence of a protective order or the receipt of a waiver hereunder, the Receiving Party is nonetheless, in the opinion of the Receiving Party's counsel, legally required to disclose the Confidential Information forwarded by the Disclosing Party or else stand liable for contempt or suffer other censure or penalty, the Receiving Party may disclose such information without

liability hereunder, provided, however, that the Receiving Party shall disclose only that portion of such Confidential Information which it is legally required to disclose. Notwithstanding anything to the contrary in the foregoing, neither party may disclose any of the Confidential Information provided by the other to any governmental bank regulatory authority having jurisdiction over said party without notice of any kind, provided (i) such disclosure is required by law or regulation, (ii) the disclosure is limited to that portion of the Confidential Information required by such bank regulatory authority to be disclosed, and (iii) such bank regulatory authority is bound by confidentiality obligations.

6. This Agreement grants no patent rights, copyrights, trade secrets or licenses, expressed or implied, to the Receiving Party except to the extent necessary for the Receiving Party to perform the evaluation contemplated by this Agreement, subject to the provisions of Paragraph 3 above.

7. Each party agrees not to export such Confidential Information, or articles incorporating the Confidential Information, to any prohibited country, as designated by the U.S. Department of Commerce, without the appropriate written authorization.

8. Each Disclosing Party warrants that it has the full right to enter into this Agreement and is the owner of its respective Confidential Information. Otherwise, no warranty, express or implied, in the Confidential Information disclosed is granted by this Agreement, and warranties of merchantability and fitness for a particular purpose are hereby disclaimed.

10. The Receiving Party acknowledges that the unauthorized disclosure of Confidential Information may cause irreparable injury to the Disclosing Party and that, in the event of a violation or threatened violation of any of Receiving Party's obligations hereunder, the Disclosing Party may have no adequate remedy at law and may therefore be entitled to seek to enforce each such obligation by temporary or permanent injunctive or mandatory relief obtained in any court of competent jurisdiction without the necessity of posting any bond or other security, and without prejudice to any other rights and remedies which may be available at law or in equity subject to the provisions of this Agreement.

11. The Receiving Party acknowledges that its obligations under this Agreement with regard to trade secrets of the Disclosing Party remain in effect for as long as such information shall remain a trade secret under applicable law. The obligations and restrictions under this Agreement shall otherwise extend for five (5) years from and after the date of the generation or disclosure of the Confidential Information, subject to the provisions of Paragraph 4 above.

12. Neither this Agreement nor any rights or obligations hereunder may be assigned by either party hereto without the prior written consent of the other. This Agreement shall inure to the benefit of and be binding upon the Parties hereto and their respective successors and assigns.

13. No delay or omission by either party hereto to exercise any right or power occurring upon any noncompliance or default by the other party with respect to any of the terms of this Agreement shall impair any such right or power or be construed to be a waiver thereof. A

waiver by either of the Parties hereto of any of the covenants, conditions, or agreements to be performed by the other shall not be construed to be a waiver of any succeeding breach thereof or of any covenant, condition, or agreement herein contained. Unless stated otherwise, all remedies provided for in this Agreement shall be cumulative and in addition to and not in lieu of any other remedies available to either party at law, in equity, or otherwise.

14. If any term or provision of this Agreement should be declared invalid by a court of competent jurisdiction, the remaining terms and provisions of this Agreement shall remain unimpaired and in full force and effect.

15. This Agreement may not be amended, modified or waived in any manner, except in writing signed by the Parties. This Agreement embodies the entire understanding between the Parties pertaining to the subject hereof. There are no prior representations, warranties, or agreements between the Parties relating hereto.

16. All notices required to be given hereunder shall be in writing to the addresses set forth below in this Agreement. Notice shall be considered delivered and effective three (3) days after mailing when sent by registered or certified mail return receipt request.

17. This Agreement shall be construed, and the legal relations between the Parties determined, in accordance with the laws of the State of California. Any action brought in connection with this Agreement shall be brought in the courts of the State of California located in the County of Los Angeles, and each party hereby irrevocably consents to the jurisdiction of such courts.

IN WITNESS WHEREOF, the Parties have caused this Agreement to be executed by their duly authorized representatives who represent having the authority to bind the respective party to this Agreement.

[Company]

[Vendor]

By:

Name:

Title:

By:

Name: Sakshi Sharma

Title: Mobility Strategist

AGREEMENT

**Between
Software Developers Inc. ("SDI")
and
Jody Rogers**

EXHIBIT 3

AGREEMENT

**Between
Software Developers Inc. ("SDI")
and
Jody Rogers**

EXHIBIT 3

iOS Design Standards for SDI Apps

Design Keys

- **Deference:** The UI helps people understand and interact with the content, but never competes with it.
- **Clarity:** Text is legible at every size, icons are precise and lucid, adornments are subtle and appropriate, and a sharpened focus on functionality motivates the design.
- **Depth:** Visual layers and realistic motion impart vitality and heighten people's delight and understanding.

Interactivity is defined through the use of several gestures. These gestures are standard across the OS and should be used according to their familiar purpose:

- **Tap**
- **Drag** (horizontal or vertical, as in scrolling)
- **Flick** (a quick drag, usually for rapid navigation)
- **Swipe** (from edge, usually to reveal more options)
- **Pinch** - open and close (zoom in/out)
- **Double Tap** (zoom)
- **Press & Hold**
- **Shake** (rarely used, to reset activity)

DESIGN CHARACTERISTICS:

Negative space. Negative space makes important content and functionality more noticeable and easier to understand. Negative space makes an app look more focused and efficient.

System fonts. iOS system fonts automatically adjust letter spacing and line height so that text is easy to read and looks great at every size. Adopt Dynamic Type so your app can respond when the user chooses a different text size.

Flat, Borderless Buttons. By default, all nav bar buttons should be borderless. In content areas, a borderless button uses context, color, and a call-to-action title to indicate interactivity. A content-area button can display a thin border or tinted background that makes it distinctive.

Color. Unless color is important to the primary content of the app, or there is another specific reason to add a lot of color, color is best used sparsely as an enhancement. A key color should be used consistently to indicate interactivity.

Adaptable layout. UI responds appropriately to changes in the display environment (multiple devices, landscape layout). Support both variants of orientation (i.e. landscape with home button on right or left).

Familiarity. Elements that have similar functions should also look similar, this conforms to user expectations and increases learnability.

Content grouping. Indentation and alignment of different information groups indicate how the groups are related and make it easier for users to find specific items.

Content size. Make sure that users can understand primary content at its default size. For example, users shouldn't have to scroll horizontally to read important text, or zoom to see primary images. Control: give tappable controls a hit target of about 44 x 44 points

Pop-ups and Alerts: Reserve for delivering essential—and ideally actionable—information. An alert interrupts the user's experience and requires a tap to dismiss, so it's important for users to feel that the alert's message warrants the intrusion.

Branding. Branding should be done primarily in the app's icon, and through the app's overall color scheme. For the best user experience, you want to quietly remind users of the brand identity through your choice of font, color, and imagery, rather than by plastering logos everywhere.

Feedback. App should respond to user action with interactive elements by highlighting briefly on a tap. More robust feedback can be given through animations.

ANIMATION

Appropriate animation is important to the user experience. Animation can:

- Communicate app status and provide feedback
- Enhance the sense of direct manipulation (more intuitive)
- Help people visualize the results of their actions

Animation should be used **sparingly**, as overuse can obstruct app flow, decrease performance, and distract users from their task.

Consistency. People are accustomed to the subtle animation used in the built-in iOS apps, so remaining consistent with those animations is important. Likewise, keeping all custom animations consistent in meaning and use within the app is vital.

Navigation Icons. Use default navigation icons provided by OS, or subtle variations wherever possible. Maintain flat graphical style with custom icons.

AGREEMENT

**Between
Software Developers Inc. ("SDI")
and
Jody Rogers**

EXHIBIT 4

iOS Coding Standards for SDI Apps

Prepared for and through a review of these documents from Apple:

- The Objective-C Programming Language
- Cocoa Fundamentals Guide
- Coding Guidelines for Cocoa
- iOS App Programming Guide

Table of Contents

- Dot-Notation Syntax
- Spacing
- Conditionals
- Ternary Operator
- Error handling
- Methods
- Variables
- Naming
- Comments
- init and dealloc
- Literals
- CGRect Functions
- Constants
- Enumerated Types
- Bitmasks
- Private Properties
- Image Naming
- Booleans
- Singletons
- Imports
- Xcode project

Dot-Notation Syntax

Dot-notation should **always** be used for accessing and mutating properties. Bracket notation is preferred in all other instances.

For example:

```
view.backgroundColor = [UIColor orangeColor];  
[UIApplication sharedApplication].delegate;
```

Not:

```
[view setBackgroundColor:[UIColor orangeColor]];  
UIApplication.sharedApplication.delegate;
```

Spacing

- Indent using 4 spaces. Never indent with tabs. Be sure to set this preference in Xcode.
- Method braces and other braces (if/else/switch/while etc.) always open on the same line as the statement but close on a new line.

For example:

```
if (user.isHappy) {  
    // Do something  
}  
else {  
    // Do something else  
}
```


- There should be exactly one blank line between methods to aid in visual clarity and organization. Whitespace within methods should separate functionality, but often there should probably be new methods.
- `@synthesize` and `@dynamic` should each be declared on new lines in the implementation.

Conditionals

Conditional bodies should always use braces even when a conditional body could be written without braces (e.g., it is one line only) to prevent errors. These errors include adding a second line and expecting it to be part of the if-statement. Another, even more dangerous defect may happen where the line "inside" the if-statement is commented out, and the next line unwittingly becomes part of the if-statement. In addition, this style is more consistent with all other conditionals, and therefore more easily scannable.

For example:

```
if (!error) {  
    return success;  
}
```

Not:

```
if (!error)  
    return success;
```

or

```
if (!error) return success;
```


Ternary Operator

The Ternary operator, `?`, should only be used when it increases clarity or code neatness. A single condition is usually all that should be evaluated. Evaluating multiple conditions is usually more understandable as an if statement, or refactored into instance variables.

For example:

```
result = a > b ? x : y;
```

Not:

```
result = a > b ? x = c > d ? c : d : y;
```

Error handling

When methods return an error parameter by reference, switch on the returned value, not the error variable.

For example:

```
NSError *error;  
if (![self trySomethingWithError:&error]) {  
    // Handle Error  
}
```

Not:

```
NSError *error;  
[self trySomethingWithError:&error];  
if (error) {  
    // Handle Error  
}
```


Some of Apple's APIs write garbage values to the error parameter (if non-NULL) in successful cases, so switching on the error can cause false negatives (and subsequently crash).

Methods

In method signatures, there should be a space after the scope (-/+ symbol). There should be a space between the method segments.

For Example:

```
- (void)setExampleText:(NSString *)text image:(UIImage *)image;
```

Variables

Variables should be named as descriptively as possible. Single letter variable names should be avoided except in `for()` loops.

Asterisks indicating pointers belong with the variable, e.g., `NSString *text` not `NSString* text` or `NSString * text`, except in the case of constants.

Property definitions should be used in place of naked instance variables whenever possible. Direct instance variable access should be avoided except in initializer methods (`init`, `initWithCoder:`, etc...), `dealloc` methods and within custom setters and getters. For more information on using Accessor Methods in Initializer Methods and `dealloc`, see [here](#).

For example:

```
@interface NYTSection: NSObject
```

```
@property (nonatomic) NSString *headline;
```


@end

Not:

```
@interface NYTSection : NSObject {  
    NSString *headline;  
}
```

Variable Qualifiers

When it comes to the variable qualifiers introduced with ARC, the qualifier (`__strong`, `__weak`, `__unsafe_unretained`, `__autoreleasing`) should be placed between the asterisks and the variable name, e.g., `NSString * __weak text`.

Naming

Apple naming conventions should be adhered to wherever possible, especially those related to memory management rules (NARC).

Long, descriptive method and variable names are good.

For example:

```
UIButton *settingsButton;
```

Not

```
UIButton *setBut;
```

A three letter prefix (e.g. NYT) should always be used for class names and constants, however may be omitted for Core Data entity names. Constants should be camel-case with all words capitalized and prefixed by the related class name for clarity.

For example:

```
static const NSTimeInterval NYTArticleViewControllerNavigationFadeAnimationDuration =  
0.3;
```

Not:

```
static const NSTimeInterval fadetime = 1.7;
```

Properties and local variables should be camel-case with the leading word being lowercase.

Instance variables should be camel-case with the leading word being lowercase, and should be prefixed with an underscore. This is consistent with instance variables synthesized automatically by LLVM. **If LLVM can synthesize the variable automatically, then let it.**

For example:

```
@synthesize descriptiveVariableName = _descriptiveVariableName;
```

Not:

```
id varnm;
```

Comments

When they are needed, comments should be used to explain **why** a particular piece of code does something. Any comments that are used must be kept up-to-date or deleted.

Block comments should generally be avoided, as code should be as self-documenting as possible, with only the need for intermittent, few-line explanations. This does not apply to those comments used to generate documentation.

init and dealloc

dealloc methods should be placed at the top of the implementation, directly after the @synthesize and @dynamic statements. init should be placed directly below the dealloc methods of any class.

init methods should be structured like this:

```
- (instancetype)init {
    self = [super init]; // or call the designated initializer
    if (self) {
        // custom initialization
    }

    return self;
}
```

Literals

NSString, NSDictionary, NSArray, and NSNumber literals should be used whenever creating immutable instances of those objects. Pay special care that nil values not be passed into NSArray and NSDictionary literals, as this will cause a crash.

For example:

```
NSArray *names = @[@"Brian", @"Matt", @"Chris", @"Alex", @"Steve", @"Paul"];
NSDictionary *productManagers = @{@"iPhone" : @"Kate", @"iPad" : @"Kamal", @"Mobile
Web" : @"Bill"};
NSNumber *shouldUseLiterals = @YES;
NSNumber *buildingZIPCode = @10018;
```

Not:


```
NSArray *names = [NSArray arrayWithObjects:@"Brian", @"Matt", @"Chris", @"Alex",
@"Steve", @"Paul", nil];
NSDictionary *productManagers = [NSDictionary dictionaryWithObjectsAndKeys: @"Kate",
@"iPhone", @"Kamal", @"iPad", @"Bill", @"Mobile Web", nil];
NSNumber *shouldUseLiterals = [NSNumber numberWithBool:YES];
NSNumber *buildingZipCode = [NSNumber numberWithInt:10018];
```

CGRect Functions

When accessing the x, y, width, or height of a CGRect, always use the CGGeometry functions instead of direct struct member access. From Apple's CGGeometry reference:

All functions described in this reference that take CGRect data structures as inputs implicitly standardize those rectangles before calculating their results. For this reason, your applications should avoid directly reading and writing the data stored in the CGRect data structure. Instead, use the functions described here to manipulate rectangles and to retrieve their characteristics.

For example:

```
CGRect frame = self.view.frame;

CGFloat x = CGRectGetMinX(frame);
CGFloat y = CGRectGetMinY(frame);
CGFloat width = CGRectGetWidth(frame);
CGFloat height = CGRectGetHeight(frame);
```

Not:

```
CGRect frame = self.view.frame;

CGFloat x = frame.origin.x;
CGFloat y = frame.origin.y;
```



```
CGFloat width = frame.size.width;
CGFloat height = frame.size.height;
```

Constants

Constants are preferred over in-line string literals or numbers, as they allow for easy reproduction of commonly used variables and can be quickly changed without the need for find and replace. Constants should be declared as `static` constants and not `#defines` unless explicitly being used as a macro.

For example:

```
static NSString * const SDIAboutViewControllerCompanyName = @"Software Developers Inc";

static const CGFloat SDIImageThumbnailHeight = 50.0;
```

Not:

```
#define CompanyName @"Software Developers Inc"

#define thumbnailHeight 2
```

Enumerated Types

When using `enums`, it is recommended to use the new fixed underlying type specification because it has stronger type checking and code completion. The SDK now includes a macro to facilitate and encourage use of fixed underlying types — `NS_ENUM()`

Example:

```
typedef NS_ENUM(NSInteger, NYTAdRequestState) {
    NYTAdRequestStateInactive,
```



```
NYTAdRequestStateLoading
};
```

Bitmasks

When working with bitmasks, use the `NS_OPTIONS` macro.

Example:

```
typedef NS_OPTIONS(NSUInteger, NYTAdCategory) {
    NYTAdCategoryAutos      = 1 << 0,
    NYTAdCategoryJobs       = 1 << 1,
    NYTAdCategoryRealState  = 1 << 2,
    NYTAdCategoryTechnology = 1 << 3
};
```

Private Properties

Private properties should be declared in class extensions (anonymous categories) in the implementation file of a class. Named categories (such as `NYTPrivate` or `private`) should never be used unless extending another class.

For example:

```
@interface NYTAdvertisement ()

@property (nonatomic, strong) GADBannerView *googleAdView;
@property (nonatomic, strong) ADBannerView *iAdView;
@property (nonatomic, strong) UIWebView *adxWebView;

@end
```


Image Naming

Image names should be named consistently to preserve organization and developer sanity. They should be named as one camel case string with a description of their purpose, followed by the un-prefixed name of the class or property they are customizing (if there is one), followed by a further description of color and/or placement, and finally their state.

For example:

- RefreshBarItem / RefreshBarItem@2x and RefreshBarItemSelected / RefreshBarItemSelected@2x
- ArticleNavigationBarWhite / ArticleNavigationBarWhite@2x and ArticleNavigationBarBlackSelected / ArticleNavigationBarBlackSelected@2x.

Images that are used for a similar purpose should be grouped in respective groups in an Images folder.

Booleans

Since `nil` resolves to `NO` it is unnecessary to compare it in conditions. Never compare something directly to `YES`, because `YES` is defined to 1 and a `BOOL` can be up to 8 bits.

This allows for more consistency across files and greater visual clarity.

For example:

```
if (!someObject) {  
}
```

Not:


```
if (someObject == nil) {  
}
```

For a BOOL, here are two examples:

```
if (isAwesome)  
if (![someObject boolValue])
```

Not:

```
if (isAwesome == YES) // Not a good idea.  
if ([someObject boolValue] == NO)
```

If the name of a BOOL property is expressed as an adjective, the property can omit the "is" prefix but specifies the conventional name for the get accessor, for example:

```
@property (assign, getter=isEditable) BOOL editable;
```

Text and example taken from the Cocoa Naming Guidelines.

Singletons

Singleton objects should use a thread-safe pattern for creating their shared instance.

```
+ (instancetype)sharedInstance {  
    static id sharedInstance = nil;  
  
    static dispatch_once_t onceToken;  
    dispatch_once(&onceToken, ^{  
        sharedInstance = [[self alloc] init];  
    });  
}
```



```
    return sharedInstance;  
}
```

This will prevent possible and sometimes prolific crashes.

Imports

If there is more than one import statement, group the statements together. Commenting each group is optional.

Note: For modules use the `@import` syntax.

```
// Modules  
@import QuartzCore;  
  
// Headers  
#import "NYTUser.h"  
  
// Views  
#import "NYTButton.h"  
#import "NYTUIView.h"
```

Xcode project

The physical files should be kept in sync with the Xcode project files in order to avoid file sprawl. Any Xcode groups created should be reflected by folders in the filesystem. Code should be grouped not only by type, but also by feature for greater clarity.

When possible, always turn on "Treat Warnings as Errors" in the target's Build Settings and enable as many additional warnings as possible. If you need to ignore a specific warning, use Clang's pragma feature.