Reminder

Colin Dunn

### Table of Contents

[Table of Contents](#_7vbf9ja7xyoi)

[Mission Statement](#_j2qiseiwi40z)

[Database](#_xo49dqo7h1vn)

[Features](#_pamvuvex3ump)

[Trello Link](#_fcmgtowcu1fr)

[Wireframe](#_q85md16hzpog)

[Schedule](#_ye3wz1kg9ovv)

[Final Notes](#_9561h8mlbskk)

### 

### 

### Mission Statement

Reminder is a social Web site that allows you to keep track of special occasions and important events for your family and friends. Never again forget a birthday or anniversary, and never miss another important event! Users can sign up, define relationships with other users, and keep track of important dates.

### Database

This is where your team writes out how your database will look like. List out each table, the columns (include the data types), and the database associations your project will have on the server-side.

Database diagram link: <https://dbdiagram.io/d/61f0964c7cf3fc0e7c67803c>

Table 1 - User Profile

| **Property** | **Type** |
| --- | --- |
| Id | Int (primary key) |
| OwnerId | GUID |
| FirstName | String |
| MiddleName | String |
| LastName | String |
| Gender | Enum (male, female, other) |
| BirthDate | DateTime |
| Photo | Byte[] |
| Email | String |
| Phone | String |
| Address | String |
| City | String |
| State | String (2 characters) |
| ZIP | String (10 characters) |

Table 2 - Relationships

| **Property** | **Type** |
| --- | --- |
| Id | Int (primary key) |
| User | GUID |
| RelatedUser | Int (foreign key) |
| HowRelated | Enum { spouse, significantOther, grandparent, parent, child, grandchild, nieceNephew, auntUncle, cousin, friend, colleague } |
| Connected | Bool |

Table 3 - Events

| **Property** | **Type** |
| --- | --- |
| Id | Int (primary key) |
| RelationshipId | Int (foreign key) |
| Date | DateTime |
| Description | String |
| NotifyBefore | DateTimeOffset |

Optional:

Additional Table - Messages

| **Property** | **Type** |
| --- | --- |
| Id | Int (primary key) |
| RelationshipId | Int (foreign key) |
| WhenSent | DateTime |
| Subject | String |
| MessageText | String |
| Image | byte[] |
| WhenRead | DateTime? |

Endpoints

* User
  + Create (register) user
  + Read user
  + Update user
  + Delete user
* Relationship
  + Create relationship
  + Read (display) relationships
  + Update relationship
  + Delete relationship
* Event
  + Create event
  + Read events
  + Update event
  + Delete event
* Message
  + Create (send) a message
  + Read (display) a message
  + Update a message
  + Delete a message

### Features

Features are instances or examples of different pieces of functionality. This is where your team lists out the features you are planning on implementing. Consider the different steps and logic those features require to do the expected job. This could include fetching data from a 3rd party API or simply looping over data from your server. Differentiate between your version 1.0 or MVP (minimal viable product) and version 2.0 or stretch goals.

| Version 1.0 / MVP | Version 2.0 / Stretch Goals |
| --- | --- |
| * Maintain user profiles for every user * Landing page shows upcoming events for the currently logged on user * “My Relationships” page has links for view / edit / delete * “My Events” page has links for view / edit / delete * CRUD for user object * CRUD for relationship object * CRUD for event object * Ability to search for another user by name * Styling of views for landing page and CRUD functions | * Ability to send a message / greeting to your connections * Ability to notify users of upcoming events via E-mail or text message * Upgrade project to use Bootstrap 5 instead of 3 * Ingest information about a user from another social media site * Enhance search function to search by location / proximity * Security for users to approve when a relationship is defined involving them. |

### 

### Trello Link

Once you have your features listed out, start writing out the tickets that each feature includes on your team’s Trello Board.

Link to Trello board: [Reminder | Trello](https://trello.com/b/w7yM2iP0/reminder)

### Schedule

This project is the equivalent of one sprint in the agile methodology. In this section, write out a schedule spanning over the next couple of weeks. This should include deployment, time set aside to tackle especially challenging features, testing, etc. Consider whether your team will be working over the weekend(s). The table below is a guideline. It is not necessary to specify each day’s work/logic. Feel free to estimate your time.

| **Sunday 01/23** | **Monday 01/24** | **Tuesday 01/25** | **Wednesday 01/26** | **Thursday 01/27** | **Friday 01/28** | **Saturday 01/29** |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | Project planning | Project planning | Project approval | Scaffold project | Saturday class |
| **Sunday 01/30** | **Monday 01/31** | **Tuesday 02/01** | **Wednesday 02/02** | **Thursday 02/03** | **Friday 02/04** | **Saturday 02/05** |
|  | Data and models layers | User Profile CRUD | Relationship CRUD | Event CRUD | View styling |  |
| **Sunday 02/06** | **Monday 02/07** | **Tuesday 02/08** | **Wednesday 02/09** | Thursday 02/10 | Friday 02/11 | Saturday 02/12 |
|  | View styling | Test user, relationship, and event CRUD | Implement search functionality | Implement message functionality | Stretch goals / debugging |  |
| Sunday 02/13 | Monday 02/14 | Tuesday 02/15 | Wednesday 02/16 | Thursday 02/17 | Friday 02/18 | Saturday 02/19 |
|  | Stretch goals / debugging | Stretch goals / debugging | Stretch goals / debugging | Complete README, update project documents, create slides, and deploy to Azure | Rehearse demo | Saturday class  Present project |

### Final Notes

Great job with planning! You are now set to start coding. Planning a project is incredibly beneficial to the success of your team and your project. Here are some resources to help you with your planning.

* [How to plan a web application](https://selftaughtcoders.com/plan-web-application/)
* [Step By Step: Planning a web application](https://medium.com/@ericwindmill/step-by-step-planning-a-web-application-ddaa010a8353)