Bethsaida

Downtown Daily Bread

Andy Guenin and Thomas Zinckgraf

V0.1

Table of Contents

[Background 2](#_Toc8722309)

[Prior Art 2](#_Toc8722310)

[Required Features and Workflows 2](#_Toc8722311)

[Bethsaida 2](#_Toc8722312)

[Timeline 3](#_Toc8722313)

[Technical Specifications 3](#_Toc8722314)

[Environments 3](#_Toc8722315)

[User’s Guide 3](#_Toc8722316)

[Encryption, Authentication, and Authorization 3](#_Toc8722317)

[Data Model 4](#_Toc8722318)

[User 4](#_Toc8722319)

[Client 4](#_Toc8722320)

[Service 5](#_Toc8722321)

## Background

Downtown Daily Bread is a mission of Pine Street Presbyterian Church in Harrisburg, Pennsylvania. The mission of Downtown Daily Bread is [to provide services to the homeless in Harrisburg, including meals, a day shelter, a night shelter, counseling, among others]. In support of this mission, Downtown Daily Bread would like method to track the usage of the services.

## Prior Art

Downtown Daily Bread currently uses an Access database to track service usage by clients. Access is a program created by Microsoft and included in their Windows version of Office. There are quite a few problems with using Access to track client usage:

1. The Access database is stored on a shared Dropbox drive. Employees of Downtown Daily Bread have had issues where concurrent access has caused merging conflicts in the data, meaning a lot of manual clean-up of the corrupt data.
2. Access does not provide an app for Apple computers. Given that some of the staff have Apple computers, this presents a problem when trying to view the database.
3. The data model in Access has not been fully fleshed out, meaning that there is a lot of duplicate information, orphan records, and incomplete records.
4. There is no permission model. There is a desire to protect certain records to be viewed by specific employees, and it is unclear whether or not this is possible to do in Access.

## Required Features and Workflows

Based on the current feature set and the shortcomings of the current system, we have compiled a list of features that will be required in this new version of a client tracking system.

* A system of authentication and authorization, where employees can log into the system, will be created. Based on the permissions that have been granted to them by the site administrator, users will have access to various components of the system.
* Users will have information collected, which will be entered into the system, where they will be onboarded.
* A set of services will be created, and users will be associated with these services for specific dates.
* A set of notes, with fine-grained permission control, will be able to be associated with clients and their association with services.
* A set of reports will be specified and generated on a periodic basis.

## Bethsaida

Bethsaida is the name that we have given to the software to keep track of clients. According to Luke 9:10, Bethsaida was the location where Jesus fed a crowd of five thousand from just five loaves of bread and two fish. Bethsaida is a replacement for the current Microsoft Access and Dropbox system to keep track of clients of Downtown Daily Bread.

## Timeline

The current timeline is to deliver the pre-release 1.0 version for user acceptance testing (UAT) by June 30. This delivery will include access to a Quality Assurance (QA) environment running the software, a user guide, and a list of features that have been completed for this deadline.

## Technical Specifications

Bethsaida is an open-source web application built from two separate, but intertwined, components. There is a web-based front-end built-in HTML, CSS, and JavaScript using the popular web framework Angular. The front-end application communicates with the back-end data service, which is written in Scala using the akka-http framework and which stores data in a PostgreSQL database. The back-end and front-end are contained in a docker container that exposes an nginx web server that reverse proxies api requests to the back-end service and serves the static front-end files. The docker container is run on Elastic Container Service (ECS) the Amazon Web Service (AWS) cloud. The PostgreSQL database is Amazon RDS. The database is periodically backed up and stored off site from the AWS cloud.

### Environments

There are two environments, a production environment and a QA environment. Code changes are automatically deployed to the QA environment, allowing for users to test before signing off on required changes. The production database is snapshotted and copied to the QA environment once a week. The QA environment contains a distinctive color scheme to indicate to users that data inserted there will not be saved for more than a week.

## User’s Guide

This user’s guide will give you a detailed description of how the site works. It discusses the data model first, and then reviews the user interface (UI), which simply is the friendly way of interacting with the data model.

### Encryption, Authentication, and Authorization

Bethsaida uses industry standard TSL encryption for http encryption. Data at rest in the database is not encrypted, but standard security practices were taken in order to secure access to the machines storing the database.

Authentication is the process of a machine determining you are who you say you are. In modern web applications, this is done by a username and password, and in Bethsaida this is no exception. The web UI will ask for these access credentials and will then make a request to the Bethsaida server, which will verify the correctness of these credentials, and will return a token to the front end. This token is used for all further requests. All access to endpoints, other than the login and reset password endpoints, will required this authentication token.

Authorization is the process of a machine determining that, given you are who you say you are, that you are allowed access to specific resources. For instance, user A should not be allowed to change user B’s password. Bethsaida has an authorization model that is discussed later.

### Data Model

At its heart, the point of Bethsaida is to better organize the data that is collected by the daily operations of Downtown Daily Bread. Data here is organized into *objects*, which are connected to each other through *relations*. The following are the objects that are stored in Bethsaida, and their relations to other objects.

|  |  |  |
| --- | --- | --- |
| object | description | relations |
| User | Represents someone who logs into the Bethsaida application. |  |
| Client | Someone who uses one or more services at Downtown Daily Bread. |  |
| Service | A service provided by Downtown Daily Bread to serve clients. A service Is expected to be provided multiple times and is not usually a one-off occurrence. |  |
| Event | An event is the occurrence of a service at a specific point in time. | *Belongs to* Service |
| Schedule | Automatically creates events for services on a given schedule. | *Belongs to* Service |
| Invitee | Associates a client to an event, where the client is expected to appear at the event. | *Belongs to* Event  *Belongs to* Client |
| Attendance | Associates a client to an event, where the client actually showed up for the event. | *Belongs to* Event  *Belongs to* Client |
| Note | Adds a text description to any of the above models that will be saved | *Belongs to* all |
| Ban | Indicates that a client has been banned from all services for a specific amount of time. | *Belongs to* client |

### User

A user is an account stored on the Bethsaida backend that allows an employee to log in to the system to view and make changes to the data. A user has a full name, an email address that is used as a username, and a password. Additionally, a user can initiate a password reset. All actions that change data are tagged to a specific user.

### Client

A client is someone who makes use of the services provided by Downtown Daily Bread. The Bethsaida administrator will set up a series of attributes to be collected by each client. These attributes fall into one of three types:

1. Required for entry into system
2. Required for onboarding
3. Optional

#### Required for entry into system

Required for entry into system will prevent the client record from being saved if all attributes of this type are not completed. This list should be short and include only the core pieces of information required for client identification. Attributes of this type can be updated at a later time, but it is strongly advised that these attributes are collected correctly the first time. These attributes are meant to be quickly processed by employees greeting clients in real time, meaning there should be no need for a sign on sheet where handwriting readability can cause an issue.

#### Required for onboarding

These attributes are required in order for a client to have been considered onboarded. These are attributes that Downtown Daily Bread still needs to collect about a client, but can come back to after everyone has been checked in. A report about clients not fully onboarded will be generated daily.

#### Optional

This list of attributes is self-explanatory. They are nice to have pieces of information about a client, but not necessary.

There is no default set of attributes for any of the above categories, giving the administrator full control of the attributes collected. If an attribute is added in either of the required fields after a client has been entered and onboarded, a notification should pop up the next time the client is checked into an Event that their attributes are not complete. While this can be ignored, it is advisable to collect the data when the alert pops up. Client attributes cannot be changed unless this missing attribute is also fixed at the same time, or the system will reject the client attributes as incomplete.

### Service

A Service is the representation of a specific fulfillment of the mission of Downtown Daily Bread. A service should be a generic description of an event, such as “One-on-one Counseling”, and not something specific such as “Review Section 8 Housing Procedures with Brad on August 5”. This more descriptive title will be customized for their specific purpose through Events.

### Events

Events are specific instances of Services on specific days for a specific begin and end time. Events cannot last more than 24 hours. Events can be automatically created through a schedule attached to a service, or manually for ad-hoc purposes. Events can have their own title.

### Invitee

Clients can be invited to specific events. Similar to outlook scheduling, clients can either be listed as required to attend or optional to attend. This is used for tracking and reporting purposes.

### Schedule

A schedule is an automated way to create an event so that it doesn’t have to be manually done. It is important to understand how this event creation works behind the scenes. Every 15 minutes, a job runs in the server that determines if any events are supposed to begin within 2 hours, according to the schedule. If it detects that one is supposed to start in 2 hours, it schedules a job to create the event 1 hour before it is supposed to start. At exactly one hour before the event is scheduled to begin, an Event is created, and client check in can begin for it.