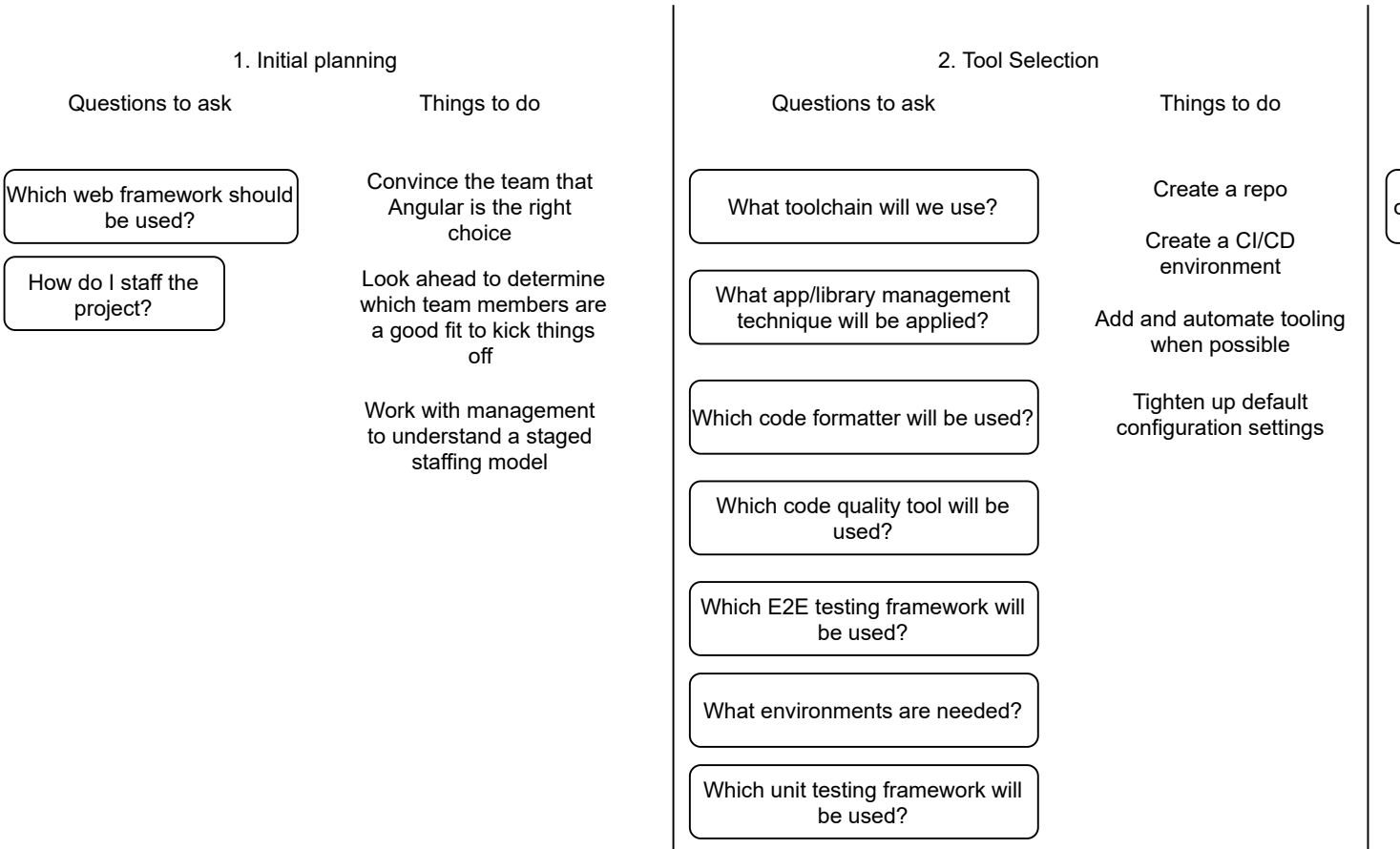
# Angular for Architects: Architectural Lifecycle

Paul Spears - @TheEvergreenDev - Oasis Digital Solutions Inc.

At Oasis Digital we often consult for CTO's, product owners, and project architects to help establish company and project best practic
From those efforts, we have created a process that helps us consistently and quickly identify potential challenges and decision points
and created the attached diagram as a roadmap for our Angular for Architects course.

The architectural questions and tasks of a project change as it matures. This diagram illustrates the major
phases of a project and outlines the work that is needed at each phase. At the bottom of each phase is a collection of topics that are
be used to learn more about those topics.



ces for Angular development.
s in a project's lifecycle. We have captured that process for the purposes of teaching

often addressed during that phase, as well as, what portion of the curriculum can

### 3. Initial Application Stubbing

| Questions to ask | Things to do |
| --- | --- |
| w will apps and libs be ed in practice (naming and folder structure)? | Create app and a few feature shells |

Examples

- ABC 200 - workshop to create app
- ABC 201 - app & workshop to create feature & components
- ABC 105 & 500 - Routing architecture

roject

### 4. Cross cutting application features

| Questions to ask | Things to do |
| --- | --- |
| How will the app be secured? | Implement auth/authz |
| How will the features be secured? | Create HTTP interceptors |
| How will the routes be secured? | Create services for use by route guards |
| What state management techniques will be used? | Provide initial boilerplate examples of any centralized state management solutions |
| Does I18N need to be considered? | Implement L10N, A11Y, I18N as needed |
| Does L10N need to be considered? | Add design system and component kits to the project |
| Does A11Y need to be considered? | |
| What design system is going to be used? | Add logging solutions |
| What logging is needed for the app? | Add error handling solution. Add a demo that works with logging |
| What kind of error handling is needed for the app? | Provide examples of global and component styling |
| What component sets will be used? | |
| What CSS architectural patterns will be used | |
| What environments are needed? | |

Examples

- ABC 105 & 500 - Routing architecture
- ABC 503 - Route param data loading
- ABC 503 - Services with route guards
- ABC 600 - Service state management
- ABC 601 - NGRX stage management

| Questions t |
| --- |
| What are the ma categories of the |
| What are the unit |
| What can be imp abstractly? as co |
| What kind of API is |

Examples

- ABC 2
- ABC 5
- ABC 5
- ABC 5
- ABC 4 (Deco
- ABC 3

5. Feature planning

| to ask | Things to do |
| --- | --- |
| ajor feature application? | Determine the practices to be used when reusing components and services |
| ts of reuse? | Establish how the more complex features are to be implemented |
| plemented nfiguration? | Shop for off-the-shelf solutions |
| being used? | Identify the most difficult features and begin pairing/mobbing to create POCs |

201 - Component hierarchy
503 - Route Param & data loading
502 - Query Params
504 - Resolvers
402 - Smart/View components
upling)
303 - API interaction with service

6. Feature execution

| Questions to ask | Things to do |
| --- | --- |
| How many developers do I need | Switch to sprints for the bulk of the developers/features |
| How long will it take | Provide code review assistance and continue pairing on the most difficult items |

Examples

- ABC 401, 707, 731 - Reactive forms
- ABC 814 - Template forms
- ABC 803 - Config driven dynamic form
- ABC 502 Query params - composing state management

Project

7. Optim

| Questions to ask |
| --- |
| Which areas of the application a runtime performance sensitive? |
| Is the application load time sensitive? |

Examples

- ABC 729.2 - preloa
- ABC 710, 711, 712

Project

- NA


izations

Things to do

re
?

Run performance metric checks

Analyze bundle size

ding strategies
- change detection

8. Code Publishing

Questions to ask

How will code be shared externally

Things to do

Connect to Artifactory/NPM/etc..

Examples

- NA

Project

- phase-8

Topics Include

9. Maintenance

Questions to ask

How will the application be kept up-to-date with third party code?

How will the application be kept up-to-date with API drift

E

ver

E
bug

Examples

- ABC 722-727 - Unit Testing (Ka
Jasmine)
- RxJS 400

Project

- phase-9

ital

ns Inc.

Things to do

stablish a practice for upgrading/updating

Implement schema rification tooling where appropriate

Expand test suites as gs and regressions are discovered

rma &

Topics Include

- State of Angular
- Benefits of Angular
- Unique characteristics of Angular
- Angular and Enterprise Development
- Understanding the "surface area" of an Angular project
- Understanding the low-level nature of web development

- Angular CLI - Pros and Cons
- NX - Pros and Cons
- NX - basics
- Prettier
- TSLint
- Monoliths and Monorepos
- Stricter compiler options (Angular and TypeScript)
- --strict flag when generating a new Angular application
- Integrating code quality tools into an IDE for minimal friction
- Unit tests with headless browser
- Builds - cached where possible
- E2E tests
- Evaluating options for mobile apps
- Ionic (hybrid HTML5/native)
- NativeScript (native)
- Progressive web app (HTML5)

T

- phase-3

opics Include

- Anatomy of an application
- Anatomy of a feature
- NX generators
- Enterprising scaling and conventions
- File structure and naming conventions of the Angular CLI
- Structure, maintainability, and ease of collaboration
- Building applications as a set of libraries
- Use cases and challenges of custom schematics
- Translating screen flows into a routing architecture

- ABC 502 - Query params/Form (state management)
- ABC 708 & 708.2 - Error handling/Logging
- ABC 728 - Global vs component styling

Project

- phase-4

Topics Include

- Route guards
- Interceptors vs inheritance
- Auth/Authz
- Role-based access
- forRoot and forChild
- L10N, I18N, A11Y
- Material Design and others
- Custom design systems
- Logging and error handling
- Scalable, maintainable CSS
- Micro-frontend architecture -Runtime code sharing
- Transient UI state
- Persistent data state
- State versus configuration
- Scope: component, feature, application
- Evaluating the need for a state management library
- RxJS observables as a data store
- NgRx
- NGXS
- Akita
- MobX
- Immutability: when, where, and how?
- Run time vs. build time illities
- Automated a11y browser testing

Project

- NA

Topics Include

- Strate
- Intera
- API im
- UI lay
- Growt
- Growt
- Query
- Resol
- Identif versus
- Trade librarie
- How t
- Codin
- Popul
- Popul
- Desig

e

gies for decoupling data from UI/UX
cting with a backend
mplementations and front-end impacts
out and component composition
h by adding more lines of concrete code
h by adding fewer lines of abstract code
, route, and matrix parameters
vers for simplified data loading
fying what should be done in the browser
s on a server
offs to consider when comparing
es
o handle writing your own libraries
g to an interface
ar third party control libraries
ar data grids for Angular
ning and coding for responsiveness

- phase-6

Topics Include

- The four pillars of Angular
- Angular Forms - a state abstraction for user interaction
- Strategies for multi-page-forms
- Nested form groups, form arrays, and CVAs to fill in the gaps
- Reactive forms vs. template-driven forms
- Creating config-driven dynamic forms
- Composing disparate state management solutions

Topics Include

- Change detection s
- Server-side renderi
- Speeding up Angul
- Setting size budget
- Troubleshooting wi
- Choosing preloadin
- Pure pipes
- `nx affected`
- Nx cache
- Nx Cloud
- Distributed builds
- Build once, deploy configuration using
- Efficient use of func
- Optimizing ngFor (u transforms)

strategies
ng
ar CI builds
ts
th source-map-explorer
g strategies

anywhere - Load runtime
APP_INITIALIZER
ctions in templates
use trackBy, avoid data

- Dependency management
- Single version policy
- Define relationship rules based on library type
- Nx
- Lerna
- Bazel for a multi-language monorepo
- Ad-hoc monorepo tooling
- via an artifact registry

Topics Include

- Balancing effort across types of
- Unit tests
  - Identifying what to test
  - Angular testing infrastruc etc.)
  - Frameworks and tools: K Jasmine, Jest
- E2E testing
  - Benefits relative to other testing
  - Strategies for efficient ar maintainable identificatio
  - Coordinating testers and
  - Frameworks and tools: C Protractor
- Living in the Angular ecosystem
  - Keeping up with the lates version
  - Dealing with version ske

testing

cture (TestBed,

Karma and

types of

nd
on of elements
l developers
Cypress,

st Angular

w